IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

First Named Applicant: Trelewicz	)	Art Unit: 2122
Serial No.: 09/693,090	)	Examiner: Kiss
Filed: October 20, 2000	)	**BLD9-2000-0060-US2**
For: **COMPILER FOR ENABLING MULTIPLE SIGNED INDEPENDENT DATA ELEMENTS PER REGISTER**	)	September 21, 2004 750 B STREET, Suite 3120 San Diego, CA 92101

APPEAL BRIEF

Commissioner of Patents and Trademarks
Washington, DC 20231

Dear Sir:

This brief is submitted under 35 U.S.C. §134 and is in accordance with 37 C.F.R. Parts 1, 5, 10, 11, and 41, effective September 13, 2004 and published at 69 Fed. Reg. 155 (August 2004). This brief is further to Appellant's Notice of Appeal filed herewith.

Table of Contents

(1) Real Party in Interest

The real party in interest is IBM Corp.

(2) Related Appeals/Interferences

No other appeals or interferences exist which relate to the present application or appeal.

(3) Status of Claims

Claims 1-29 are pending and finally rejected.

(4) Status of Amendments

No amendments are outstanding.

(5) Concise Explanation of Subject Matter in Each Independent Claim, with Page and Figure Nos.

As an initial matter, it is noted that according to the Patent Office, the concise explanations under this section are for Board convenience, and do not supersede what the claims actually state, 69 Fed. Reg. 155 (August 2004), see page 49976. Accordingly, nothing in this Section should be construed as an estoppel that limits the actual claim language.

Claim 1 recites a computer system that includes a compiler (11, second line from the bottom of page 6, Figure 1) receiving higher-level code and outputting lower-level code to enable a processor (12, id.) to simultaneously process multiple multi-bit data elements in a single register (24, Figure 3, page 9 at the top). The logic of the lower-level code includes establishing at least first and second signed, multi-bit data elements in at least a first register (block 56, Figure 7, last paragraph on page 17). The logic also includes simultaneously processing the elements (id.) A carry and/or a borrow may occur between data elements in a register (Figure 9, last full paragraph on page 19).

The reference numerals and page and figure numbers above are incorporated in the summary of Claim 7, which recites a computer program device that includes a computer program storage device (Figure 1, last two lines of page 8 continuing to page 9) which is readable by a digital processing apparatus and a compiler program on the program storage device. The program includes instructions that are executable by the digital processing apparatus for performing method acts for outputting lower-level code to process multi-bit, signed data elements. The lower-level code includes computer readable code means for packing at least first and second data elements into a single register, and computer readable code means for processing the elements simultaneously. The elements are permitted to interact with each other (such as by, e.g., carries and borrows).

The reference numerals and page and figure numbers above are incorporated in the summary of Claim 11, which recites a method that includes defining at least one compiler directive, instructions, or configuration file for a compiler (page 5, penultimate paragraph of summary) that defines an input precision for at least one data element and/or multiple data sources of respective data elements to be packed into a common register and operated on by an algorithm simultaneously with each other (id.) The data elements can carry from each other, and a carry propagating left without a previous borrow having occurred cannot happen and further a borrow can only follow a borrow in underflow conditions such that in any one processing cycle, at most one borrow occurs from each element in the register (page 17, top of page).

(6) Grounds of Rejection to be Reviewed on Appeal

(a) Claims 6, 9, 11-18, 21, 23, 25, 26, and 29 have been rejected under 35 U.S.C. §112, first paragraph, for lacking adequate written description.

(b) Claims 1-6, 11-21, and 23-26 have been rejected under 35 U.S.C. §102 as being anticipated by [FiD98].

(c) Claims 5 and 29 have been rejected under 35 U.S.C. §103 as being unpatentable over [FiD98] in view of Staugaard, Jr.

(d) Claims 5 and 29 have been rejected under 35 U.S.C. §103 as being unpatentable over [FiD98] in view of Mendel, USPN 6,080,204.

(7) Argument

As an initial matter, it is noted that according to the Patent Office, a new ground of rejection in an examiner's answer should be "rare", and should be levied only in response to such things as newly presented arguments by Applicant or to address a claim that the examiner previously failed to address, 69 Fed. Reg. 155 (August 2004), see, e.g., pages 49963 and 49980. Furthermore, a new ground of rejection must be approved by the Technology Center Director or designee and in any case must come accompanied with the initials of the conferees of the appeal conference, id., page 49979.

(7a) Commencing with the written description rejection which is predicated on the allegation that the amendment to the specification added new subject matter when it clarified that "input precision" means the initial precision of individual data elements ~~prior to~~ after simultaneous operation, on page 4 in the paragraph immediately following the paragraph that was amended, it is disclosed that "when the output precision is to be determined from the input precision by the compiler, the compiler counts the number and types of operations ***to be performed on the data, extending the input precision by one bit for each addition or subtraction and by sufficient bits to accommodate multiplication operations***, and by one bit when necessary to ensure that the maximum magnitude negative number that can be represented by a data element is one larger than the maximum negative number that can be represented in the output precision" (emphasis mine). Thus, as stated in this paragraph the input precision does indeed reflect the initial precision of individual data elements <u>after</u> simultaneous operation has been accounted for. It appears that the new matter objection and related written description rejections are incorrect.

(7b) The anticipation rejection is perhaps even easier to overcome, because under the examiner's own admissions as to how he is reading the claims, they are patentable over FiD98. More specifically, the examiner reasons that because the claims use permissive language (e.g., Claim 1 recites that a carry and/or a borrow "may" occur between data elements in a register), this means that the limitations are optional - which they are, in marked contrast to FiD98. That is why the inventor's prior Rule 132 declaration must not be so facilely brushed off. By its explicit terms, which the examiner readily concedes, Claim 1 requires that carries and/or borrows "may" occur, whereas in FiD98, the separate elements being processed in a single register ***are not allowed, ever*** to interact with each other by, e.g., carrying or borrowing. Instead, in the FiD98 system the elements ("fields") are isolated from each other (page 5, top, discussing partitioned operations) either by "spacer bits" (page 6, particularly last paragraph) or by using additional instructions (pages 7 and 8). Nowhere does the FiD98 paper suggest otherwise. In contrast, the present invention as now claimed makes such interaction permissible.

Thus, it is this very permissivity that the examiner admits is in the claims that makes them patentable over FiD98. There is simply no way to regard a reference in which an action is always forbidden to be the same as a claimed invention that permits the action to occur. Stated differently, in FiD98, as has been established by the Rule 132 declaration it is quite incorrect to state that a carry or borrow "may" occur, in contrast to Claim 1, when in fact such features ***can never*** occur in the FiD98 system.

(7c) Claims 5 and 29 have been rejected under 35 U.S.C. §103 as being unpatentable over FiD98 in view of Staugaard, Jr., used as a teaching of non-compiled comments with the proferred suggestion to combine being "to provide additional documentation without changing the behavior of a program". Nowhere, however, is any prior art citation to this suggestion made, and more importantly, nowhere does FiD98 suggest that providing additional documentation is a problem. That means that the requisite prior art motivation to combine Staugaard, Jr. with FiD98 is missing, because whatever reason Staugaard, Jr. gives for using comments, there has been no showing relating that reason to anything in FiD98. Likewise, Appellant respectfully asserts that no prior art reason or motivation has been pointed to for combining Mendel with FiD98.

Respectfully submitted,

[illegible]

John L. Rogitz
Registration No. 33,549
Attorney of Record
750 B Street, Suite 3120
San Diego, CA 92101
Telephone: (619) 338-8075

JLR:jg

APPENDIX A - APPEALED CLAIMS

1. A computer system, comprising:

a compiler receiving higher-level code and outputting lower-level code to enable a processor to simultaneously process multiple multi-bit data elements in a single register, the logic of the lower-level code including:

establishing at least first and second signed, multi-bit data elements in at least a first register; and

simultaneously processing the elements, wherein at least one of: a carry, and a borrow, may occur between data elements in a register.

2. The computer system of Claim 1, wherein the compiler accesses at least one of: a compiler directive, a flag, or a configuration file, to decide when to make elements independent of each other.

3. The computer system of Claim 1, wherein a first element is provided from a first data set and a second element is provided from a second data set different than the first.

4. The computer system of Claim 1, wherein the compiler allocates a respective output precision in a register for each data element to be processed in the register during a single cycle.

5. The computer system of Claim 1, wherein the compiler receives instructions not to compile a predetermined portion of code received by the compiler.

6. The computer system of Claim 1, wherein an output precision or an input precision is defined by means of a compiler directive, or a configuration file, or a variable definition.

7. A computer program device comprising:

a computer program storage device readable by a digital processing apparatus; and

a compiler program on the program storage device and including instructions executable by the digital processing apparatus for performing method acts for outputting lower-level code to process multi-bit, signed data elements, the lower-level code comprising:

computer readable code means for packing at least first and second data elements into a single register; and

computer readable code means for processing the elements simultaneously, the elements being permitted to interact with each other.

8. The computer program device of Claim 7, further comprising:

flag means indicating whether a precision should be checked in at least one cycle.

9. The computer program device of Claim 7, further comprising:

compiler directive means for defining an input precision.

10. The computer program device of Claim 7, further comprising:

compiler directive means for defining multiple data sources of respective data elements to be packed into a common register and operated on by an algorithm simultaneously with each other.

11. A method, comprising:

defining at least one compiler directive, instructions, or configuration file for a compiler defining at least one of:

an input precision for at least one data element; and

multiple data sources of respective data elements to be packed into a common register and operated on by an algorithm simultaneously with each other, wherein the data elements can carry from each other, and wherein a carry propagating left without a previous borrow having occurred cannot happen and further wherein a borrow can only follow a borrow in underflow conditions such that in any one processing cycle, at most one borrow occurs from each element in the register.

12. The method of Claim 11, wherein the compiler determines first and second precisions to be allocated in a single register to hold respective first and second signed data elements, and the compiler generates a lower-level code from a higher level code to undertake method acts comprising:

packing the elements into the register; and

operating on the elements.

13. The method of Claim 12, wherein the register sends plural data elements simultaneously to at least one computational subsystem.

14. The method of Claim 13, wherein the operation is a multiplication by a constant or by a variable of known precision, or an addition, or a shift-left logical, or a subtraction, or a bitwise AND, or a bitwise OR.

15. The method of Claim 14, wherein the elements are independent of each other as defined by the compiler directive or configuration file.

16. The method of Claim 15, wherein the first element is provided from a first data set and the second element is provided from a second data set different than the first.

17. The method of Claim 12, wherein the first element is a first partial element having a related second partial element established in a second register, and the lower-level code causes the first and second partial elements to be combined after processing.

18. The method of Claim 12, wherein the act of determining first and second precisions includes determining the precisions such that the maximum negative number that can be represented in an element is one larger than the maximum negative number that can be represented in the respective precision.

19. The computer system of Claim 2, wherein the compiler generates instructions to pack multiple data elements from respective data sources into a common register to be operated on by an algorithm simultaneously with each other.

20. The computer system of Claim 19, wherein the first element is a first partial element having a related second partial element established in a second register, and the lower-level code output by the compiler causes the first and second partial elements to be combined after processing.

21. The method of Claim 11, wherein the compiler directive, instructions, or configuration file embodies instructions to compile predetermined portions of code received by the compiler to be executed simultaneously on packed data.

22. The computer program device of Claim 7, further comprising:

means for indicating whether a precision should be checked;

means responsive to the means for indicating for checking that the packed elements do not overflow or underflow or achieve a maximum magnitude negative number representable in the precision; and

means for, when packed elements overflow or underflow or achieve a maximum magnitude negative number representable in the precision in a cycle, undertaking wrap or saturation in the elements to prevent corruption of other data elements in a register, or signalling an error to be handled by an error-handling routine in the program.

23. The computer system of Claim 4, wherein the compiler determines the output precision based at least in part on an input precision.

24. The computer system of Claim 4, wherein the compiler receives, as input, the output precision.

25. The computer system of Claim 23, wherein the compiler adds a bit of precision if the maximum magnitude negative number that is required for the data during processing is the maximum magnitude negative number that can be represented in the respective precision.

26. The computer system of Claim 23, wherein the compiler adds at least one bit of precision based at least in part on an operation on a data element.

27. The computer program device of Claim 7, further comprising means for adding a bit of precision if the maximum magnitude negative number that is required for the data during processing is the maximum magnitude negative number that can be represented in the respective precision.

28. The computer program device of Claim 7, further comprising means for adding at least one bit of precision based at least partially on an operation on a data element.

29. The method of Claim 11, further comprising defining instructions not to compile a predetermined portion of code received by the compiler.

APPENDIX B - EVIDENCE

The Rule 132 declaration of record is in evidence.

APPENDIX C - RELATED PROCEEDINGS

None (this sheet made necessary by 69 Fed. Reg. 155 (August 2004), page 49978.)